GANDHI SCHOOL OF ENGINEERING

Chidananda Hills, Bhabandha, Berhampur, Ganjam

PPS LABORATORY EXPERIMENTS

EXPERIMENT NO. 01

Name of the Practical

Display College Name Twenty Times

Aim

To display the college name twenty times on the screen.

Algorithm

1. Start
2. Repeat a loop 20 times.
3. Display the college name in each iteration.
4. Stop

Program

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int i;
    for (i = 1; i <= 20; i++)
    {
        printf("ABC College\n");
        /* Replace with your college name */
    }
    getch();
}
```

Output

ABC College
ABC College
. . . . . (printed 20 times)

EXPERIMENT NO. 02

Name of the Practical

Display and Add All Even Numbers from 1 to 100

Aim

To display all even numbers from 1 to 100 and calculate their sum.

Algorithm

1. Start and initialize sum = 0.
2. Loop i from 2 to 100, increasing by 2.
3. Display i and add i to sum.
4. Display sum.
5. Stop

Program

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int i, sum = 0;
    printf("Even numbers: ");
    for (i = 2; i <= 100; i += 2)
    {
        printf("%d ", i);
        sum += i;
    }
    printf("\nSum = %d\n", sum);
    getch();
}
```

Output

Even numbers: 2 4 6 8 100
Sum = 2550

EXPERIMENT NO. 03

Name of the Practical

Find the Smallest and Largest Elements in an Array

Aim

To find the smallest and largest values in an array.

Algorithm

1. Read the number of elements n.
2. Read n array elements.
3. Set smallest and largest to the first element.
4. Compare each remaining element and update smallest / largest.
5. Display both values and stop.

Program

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int a[100], n, i, smallest, largest;
    printf("Enter number of elements (1-100): ");
    scanf("%d", &n);
    if (n < 1 || n > 100) return 1;
    for (i = 0; i < n; i++)
    {
        printf("Enter element %d: ", i + 1);
        scanf("%d", &a[i]);
    }
    smallest = largest = a[0];
    for (i = 1; i < n; i++)
    {
        if (a[i] < smallest) smallest = a[i];
        if (a[i] > largest) largest = a[i];
    }
    printf("Smallest = %d\nLargest = %d\n", smallest, largest);
    getch();
}
```

Output

```
Enter number of elements (1-100): 5
Enter element 1: 12
Enter element 2: 4
Enter element 3: 25
Enter element 4: 7
Enter element 5: 10
Smallest = 4
Largest = 25
```

EXPERIMENT NO. 04

Name of the Practical

Sort Array Elements in Ascending and Descending Order

Aim

To sort array elements in ascending and descending order.

Algorithm

1. Read n and the array elements.
2. Use two loops to compare each pair of elements.
3. Swap elements when they are out of order.
4. Display the array in ascending order.
5. Display it in reverse order for descending order.

Program

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int a[100], n, i, j, temp;
    printf("Enter number of elements (1-100): ");
    scanf("%d", &n);
    if (n < 1 || n > 100) return 1;
    for (i = 0; i < n; i++)
        scanf("%d", &a[i]);
    for (i = 0; i < n - 1; i++)
        for (j = 0; j < n - 1 - i; j++)
            if (a[j] > a[j + 1])
            {
                temp = a[j];
                a[j] = a[j + 1];
                a[j + 1] = temp;
            }
    printf("Ascending: ");
    for (i = 0; i < n; i++) printf("%d ", a[i]);
    printf("\nDescending: ");
    for (i = n - 1; i >= 0; i--) printf("%d ", a[i]);
    printf("\n");
    getch();
}
```

Output

Input: 5 2 9 1 4
Ascending: 1 2 4 5 9
Descending: 9 5 4 2 1

EXPERIMENT NO. 05

Name of the Practical

Enter and Display a 3 × 3 Matrix

Aim

To read the elements of a 3 × 3 matrix and display them.

Algorithm

1. Start
2. Read nine elements using nested loops.
3. Display the matrix using nested loops.
4. Stop.

Program

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int a[3][3], i, j;
    printf("Enter 9 elements of the matrix:\n");
    for (i = 0; i < 3; i++)
        for (j = 0; j < 3; j++)
            scanf("%d", &a[i][j]);
    printf("Matrix:\n");
    for (i = 0; i < 3; i++)
    {
        for (j = 0; j < 3; j++)
            printf("%d\t", a[i][j]);
        printf("\n");
    }
    getch();
}
```

Output

Input: 1 2 3 4 5 6 7 8 9
Matrix:
1 2 3
4 5 6
7 8 9

EXPERIMENT NO. 06

Name of the Practical

Addition and Subtraction of Two 3 × 3 Matrices

Aim

To calculate the addition and subtraction of two 3 × 3 matrices.

Algorithm

1. Read matrices A and B.
2. For every position, calculate sum = A + B and difference = A − B.
3. Display both result matrices.
4. Stop

Program

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int a[3][3], b[3][3], i, j;
    printf("Enter 9 elements for matrix A:\n");
    for (i = 0; i < 3; i++)
        for (j = 0; j < 3; j++) scanf("%d", &a[i][j]);
    printf("Enter 9 elements for matrix B:\n");
    for (i = 0; i < 3; i++)
        for (j = 0; j < 3; j++) scanf("%d", &b[i][j]);
    printf("A+B:\n");
    for (i = 0; i < 3; i++)
    {
        for (j = 0; j < 3; j++) printf("%d\t", a[i][j] + b[i][j]);
        printf("\n");
    }
    printf("A-B:\n");
    for (i = 0; i < 3; i++)
    {
        for (j = 0; j < 3; j++) printf("%d\t", a[i][j] - b[i][j]);
        printf("\n");
    }
    getch();
}
```

Output

For A = [[1,2,3], [4,5,6], [7,8,9]] and B = [[9,8,7], [6,5,4], [3,2,1]]:
A+B = [[10,10,10], [10,10,10], [10,10,10]]
A-B = [[-8,-6,-4], [-2,0,2], [4,6,8]]

EXPERIMENT NO. 07

Name of the Practical

Multiplication of Two 3 × 3 Matrices

Aim

To multiply two 3 × 3 matrices.

Algorithm

1. Read matrices A and B.
2. Initialize each result element C[i][j] to zero.
3. Use three loops to calculate C[i][j] += A[i][k] × B[k][j].
4. Display matrix C.
5. Stop

Program

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int a[3][3], b[3][3], c[3][3] = {0};
    int i, j, k;
    printf("Enter 9 elements for matrix A:\n");
    for (i = 0; i < 3; i++)
        for (j = 0; j < 3; j++) scanf("%d", &a[i][j]);
    printf("Enter 9 elements for matrix B:\n");
    for (i = 0; i < 3; i++)
        for (j = 0; j < 3; j++) scanf("%d", &b[i][j]);
    for (i = 0; i < 3; i++)
        for (j = 0; j < 3; j++)
            for (k = 0; k < 3; k++)
                c[i][j] += a[i][k] * b[k][j];
    printf("Product matrix:\n");
    for (i = 0; i < 3; i++)
    {
        for (j = 0; j < 3; j++) printf("%d\t", c[i][j]);
        printf("\n");
    }
    getch();
}
```

Output

A = identity matrix and B = [[1,2,3], [4,5,6], [7,8,9]]
Product matrix:
1 2 3
4 5 6
7 8 9

EXPERIMENT NO. 08

Name of the Practical

Demonstrate strlen(), strcpy(), strcat() and strcmp()

Aim

To demonstrate string functions from the standard C library.

Algorithm

1. Declare character arrays.
2. Find the length of a string with strlen().
3. Copy a string using strcpy().
4. Join strings using strcat().
5. Compare strings using strcmp() and display the result.

Program

```
#include <stdio.h>
#include <string.h>
#include <conio.h>
void main()
{
    char source[] = "Hello";
    char copy[30];
    char joined[30] = "Hello";
    int result;
    printf("Length = %zu\n", strlen(source));
    strcpy(copy, source);
    printf("Copied string = %s\n", copy);
    strcat(joined, " World");
    printf("Concatenated string = %s\n", joined);
    result = strcmp("abc", "abc");
    printf("strcmp result = %d\n", result);
    getch();
}
```

Output

```
Length = 5
Copied string = Hello
Concatenated string = Hello World
strcmp result = 0
```

Note: strcmp() returns 0 when the strings are equal; non-equal strings return a negative or positive value.

EXPERIMENT NO. 09

Name of the Practical

Calculate the Area of a Circle Using a Function

Aim

To calculate the area of a circle using a user-defined function.

Algorithm

1. Start
2. Read the radius.
3. Call the area function.
4. Calculate area = 3.14159 × radius × radius.
5. Display the area and stop.

Program

```
#include <stdio.h>
#include <conio.h>
double circleArea(double radius)
{
    return 3.14159 * radius * radius;
}
void main()
{
    double radius;
    printf("Enter radius: ");
    scanf("%lf", &radius);
    printf("Area = %.2f\n", circleArea(radius));
    getch();
}
```

Output

```
Enter radius: 5
Area = 78.54
```

EXPERIMENT NO. 10

Name of the Practical

Factorial of a Number Using Recursion

Aim

To calculate the factorial of a non-negative integer using recursion.

Algorithm

1. Read n.
2. If n is 0 or 1, return 1.
3. Otherwise, return n × factorial(n − 1).
4. Display the factorial.
5. Stop (this example accepts n from 0 to 20 for long long safety).

Program

```
#include <stdio.h>
#include <conio.h>
long long factorial(int n)
{
    if (n <= 1) return 1;
    return n * factorial(n - 1);
}
void main()
{
    int n;
    printf("Enter a number (0-20): ");
    scanf("%d", &n);
    if (n < 0 || n > 20)
    {
        printf("Please enter a number from 0 to 20.\n");
        return 1;
    }
    printf("Factorial = %lld\n", factorial(n));
    getch();
}
```

Output

```
Enter a number (0-20): 5
Factorial = 120
```

EXPERIMENT NO. 11 (A)

Name of the Practical

Call by Value

Aim

To demonstrate call by value, where a function receives copies of arguments.

Algorithm

1. Initialize a variable.
2. Pass it to a function that changes its local parameter.
3. Display the variable before and after the call.
4. Stop

Program

```
#include <stdio.h>
#include <conio.h>
void change(int x)
{
    x = 100;
}
void main()
{
    int a = 10;
    printf("Before function: %d\n", a);
    change(a);
    printf("After function: %d\n", a);
    getch();
}
```

Output

Before function: 10
After function: 10

EXPERIMENT NO. 11 (B)

Name of the Practical

Call by Reference Using Pointers

Aim

To demonstrate changing a variable through its address (pointer) in C.

Algorithm

1. Initialize a variable.
2. Pass its address to a function.
3. Change the value through the pointer.
4. Display the changed value.

Program

```
#include <stdio.h>
#include <conio.h>
void change(int *x)
{
    *x = 100;
}
void main()
{
    int a = 10;
    printf("Before function: %d\n", a);
    change(&a);
    printf("After function: %d\n", a);
    getch();
}
```

Output

Before function: 10
After function: 100

Note: C uses pass-by-value; passing a pointer allows a function to modify the original variable.

EXPERIMENT NO. 12

Name of the Practical

Arithmetic Operations Using Pointers

Aim

To perform arithmetic operations on values accessed through pointers.

Algorithm

1. Declare two integers and pointers to them.
2. Read the two integers.
3. Use the pointers to access values and calculate sum, difference, product and quotient.
4. Check that the divisor is not zero.
5. Display results.

Program

```
#include <stdio.h>
#include <conio.h>
void main()
{
    int a, b;
    int *p = &a, *q = &b;
    printf("Enter two integers: ");
    scanf("%d %d", &a, &b);
    printf("Addition = %d\n", *p + *q);
    printf("Subtraction = %d\n", *p - *q);
    printf("Multiplication = %d\n", *p * *q);
    if (*q != 0)
        printf("Division = %.2f\n", (double)*p / *q);
    else
        printf("Division is not possible (divisor is zero).\n");
    getch();
}
```

Output

```
Enter two integers: 20 5
Addition = 25
Subtraction = 15
Multiplication = 100
Division = 4.00
```